1. How do you know if a certain directory is a git repository?

You can check if there is a ".git" directory inside it.

2. How to check if a file is tracked and if not, and then track it?

3. What is the difference between git pull and git fetch?

git pull = git fetch + git merge

When you run git pull, it gets all the changes from the remote or central repository and attaches it to your corresponding branch in your local repository.

Git fetch gets all the changes from the remote repository, stores the changes in a separate branch in your local repository

4. Explain the following: git directory, working directory and staging area

The Git directory is where Git stores the Meta data and object database for your project. This is the most important part of Git, and it is what is copied when you clone a repository from another computer.

The working directory is a single checkout of one version of the project. These files are pulled out of the compressed database in the Git directory and placed on disk for you to use or modify.

The staging area is a simple file (.git/index), generally contained in your Git directory, that stores information about what will go into your next commit. It’s sometimes referred to as the index, but it’s becoming standard to refer to it as the staging area.

Generally each commit is done by 3 different directories inside objects folder.

This answer taken from git-scm.com

Refer:

<https://www.linkedin.com/pulse/git-internals-how-works-kaushik-rangadurai/>

<https://www.daolf.com/posts/git-series-part-1/>

5. How to resolve git merge conflicts?

Uses merge tools to identify the delta. Get the files fixed,

git add .

git commit -m "<comment>

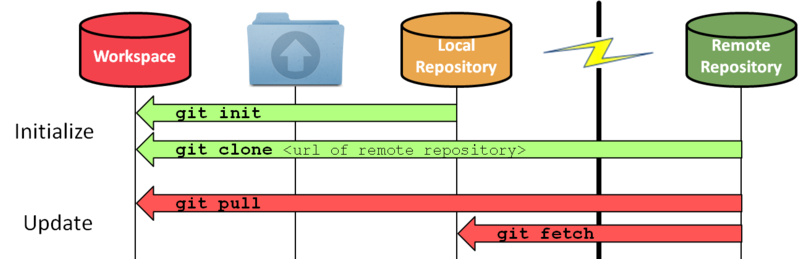
git push origin master

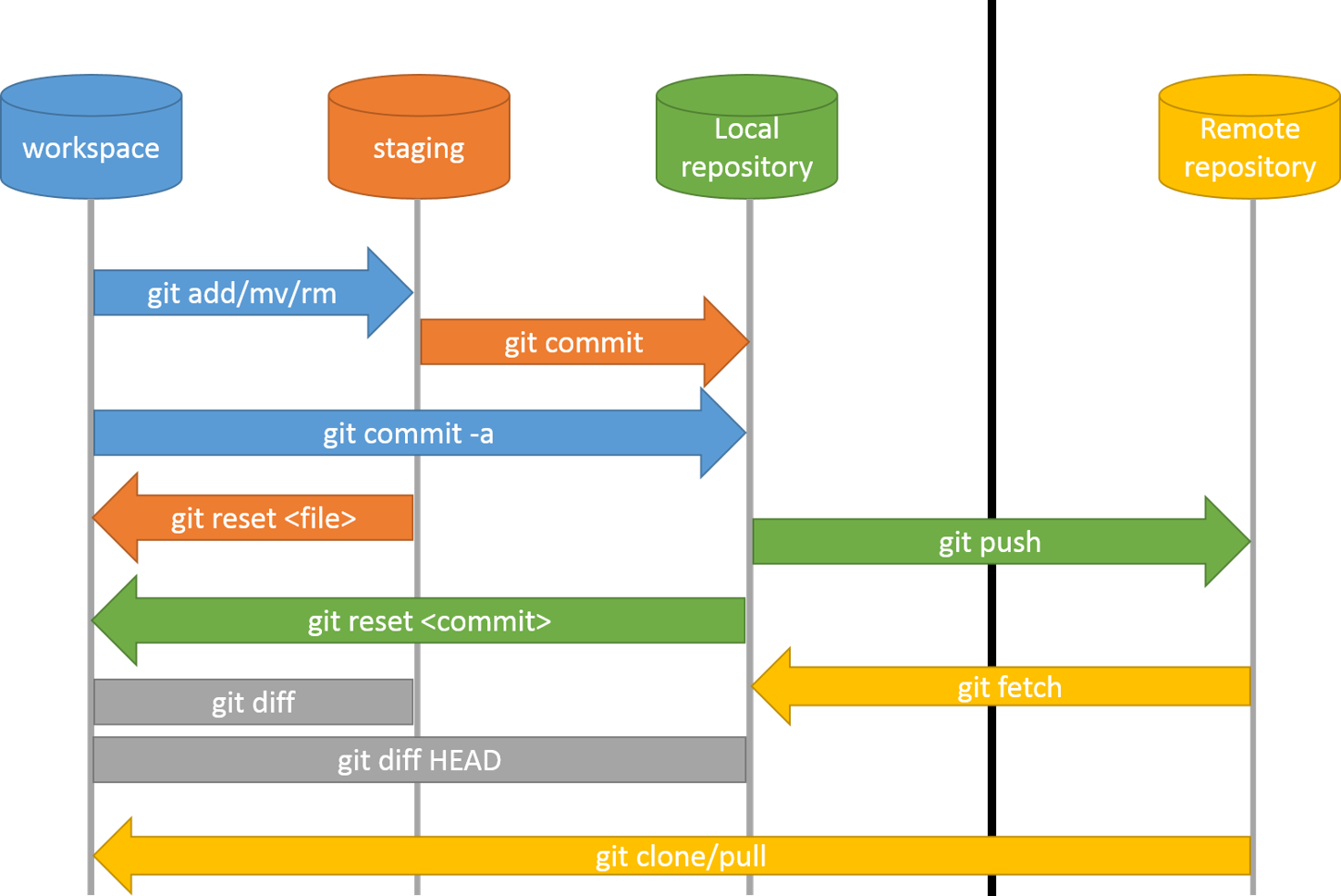
6. What is the difference between git reset and git revert?

Git revert creates a new commit which undoes the changes from last commit.

Git reset depends on the usage, can modify the index or change the commit which the branch head is currently pointing at.

7. Explain git workflow.





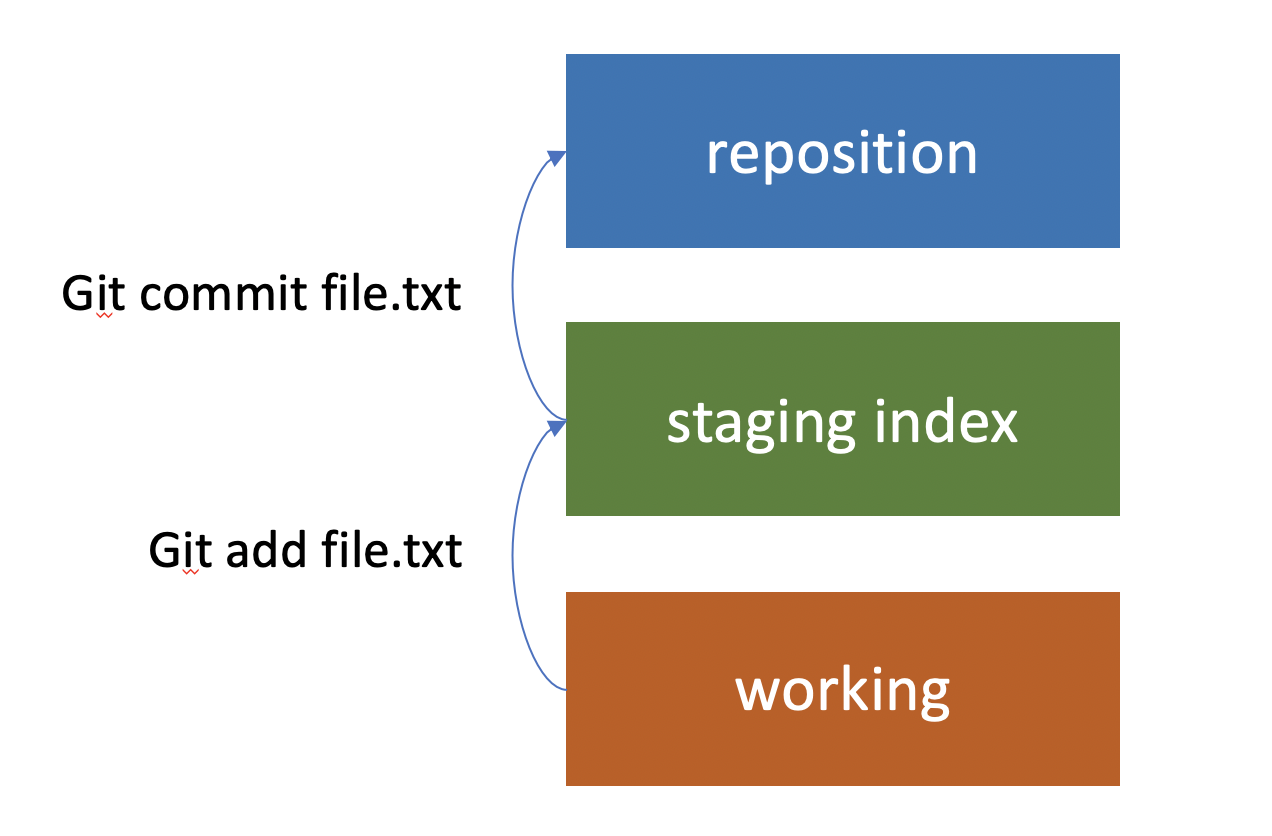
<https://www.designveloper.com/blog/git-concepts-architecture/>

8. Explain the three-trees architecture

When we made our first commit, first, we used the *add* command. We added, then we committed.It was a two-step process (added our files to the staging index, and then from there we committed to the repository).

<https://manivelarjunan.medium.com/git-merge-vs-rebase-reset-vs-revert-vs-checkout-dd5674d0e18a>

As we’re working with Git, it’s useful to keep these three different trees in mind. There’s our working directory, which contains changes that may not be tracked by Git yet, there’s the staging index, which contains changes that we’re about to commit into the repository, and then there’s the repository, and that’s what’s actually being tracked by Git.



9. You would like to move forth commit to the top. How would you achieve that?

Git rebase

10. In what situations are you using git rebase?

When there are no changes since branching in the master/source branch

11. What merge strategies are you familiar with?

<https://git-scm.com/docs/merge-strategies>

12. How can you see which changes have done before committing them?

Git diff

13. How do you revert a specific file to previous commit?

git checkout HEAD~1 -- /path/of/the/file

14. What is the .git directory? What can you find there?

The .git folder contains all the information that is necessary for your project in version control and all the information about commits, remote repository address, etc. All of them are present in this folder. It also contains a log that stores your commit history so that you can roll back to history.

<https://stackoverflow.com/questions/29217859/what-is-the-git-folder>

15. What are the various delete files conditions and how do you delete them?

rm -rf <file.name>

To remove the file from the Git repository and the file system (working dir), use:

git rm file1.txt

git commit -m "remove file1.txt"

But if you want to remove the file only from the Git repository and not remove it from the file system, use:

git rm --cached file1.txt

git commit -m "remove file1.txt"

To remove directory and content

git rm -r directory

‘- r’ can be used with all above commands to do it recursively on directories.

To remove sensitive data - Avoid as much as possible (from all commit's in history too)

<https://help.github.com/articles/remove-sensitive-data/>

To revert the previous stage

git reset

or v2.24.0+ use

git restore --staged .

16. How do you move and rename files?

Move the hello.html file to the lib directory

mkdir lib

git mv hello.html lib

git status

By moving files with git, we notify git about two things

hello.html file was deleted.

lib/hello.html file was created.

Both facts are staged immediately and ready for a commit.

Git status command reports the file has been moved.

So it is equivalent to

mkdir lib

mv hello.html lib

git add lib/hello.html

git rm hello.html

17. What are the various undo changes options?

Unstage all files you might have staged with git add:

git reset

Revert all local uncommitted changes (should be executed in repo root):

git checkout .

Revert uncommitted changes only to particular file or directory:

git checkout [some\_dir|file.txt]

Revert all uncommitted changes (longer to type, but works from any subdirectory):

git reset --hard HEAD

Remove all local untracked files, so only git tracked files remain:

git clean -fdx

WARNING: -x will also remove all ignored files, including ones specified by .gitignore! You may want to use -n for preview of files to be deleted.

Nothing in Git is deleted, so when you work in Git, you can undo your work.

When you can undo changes

In the standard Git workflow:

Untracked files: New file created

Unstaged files: Edit a staged file.

Untracked is also a unstaged file.

Index files/Staged files:

git add, git mv etc.

Committed file

(git commit).

File pushed to remote repo.

(git push).

#Undo unstaged local changes

----------------------------

Confirm that the file is unstaged (that you did not use git add <file>)

$ git status

To overwrite local changes:

git checkout -- <file>

To save local changes so you can re-use them later:

git stash

To discard local changes to all files, permanently:

git reset --hard

#Undo staged local changes

--------------------------

Confirm that the file is staged (that you used git add <file>)

$ git status

To unstage the file but keep your changes:

git restore --staged <file>

To unstage everything but keep your changes:

git reset

To unstage the file to current commit (HEAD):

git reset HEAD <file>

To discard all local changes, but save them for later:

git stash

To discard everything permanently:

git reset --hard

#Quickly save local changes

---------------------------

If you want to change to another branch

git stash.

From the branch where you want to save your work,

git stash.

Swap to another branch (git checkout <branchname>).

Commit, push, and test.

Return to the branch where you want to resume your changes.

Use git stash list to list all previously stashed commits.

Run a version of git stash:

Use git stash pop to redo previously stashed changes and remove them from stashed list.

Use git stash apply to redo previously stashed changes, but keep them on stashed list.

<https://docs.gitlab.com/ee/topics/git/numerous_undo_possibilities_in_git/>

18. How do you retrieve old versions?

Restore working dir to the latest in remote repo.

$ git checkout HEAD filename

$ git log --oneline

79a4e5f bad take

f449007 The second commit

55df4c2 My great project, first commit.

$ git checkout 55df4c2 filename

19. Differerence between revert, reset and rebase

<https://opensource.com/article/18/6/git-reset-revert-rebase-commands>

20. Differerence between soft reset, mixed reset and hard reset

<https://davidzych.com/difference-between-git-reset-soft-mixed-and-hard/>

21. How do you remove untracked files?

Define them in .gitignore

If you have added new files, before running the actual command and removing untracked files and directories use the -n option that will perform a “dry run” and show you what files and directories will be deleted:

git clean -d -n

To delete the file

git clean -d -f

22. Explain git branching

<https://www.javatpoint.com/git-branch>

23. How do you Switch branches?

git checkout <branch>

24. How do you Create and switch branches?

git checkout –b <branch>

25. How do you Switch branches with uncommitted changes?

Always add, commit or stash your changes before switching

26. How do you Compare branches?

git diff <branch1> <branch2>

27. How do you Rename branches?

git branch –m <old name> <new name>

<https://linuxize.com/post/how-to-rename-local-and-remote-git-branch/>

28. How do you Delete branches?

Delete local branches

Delete merged branch

git branch -d <branch>

Delete unmerged branch

git branch -D <branch>

Delete remote branch

git push <remote> --delete <branch>

git push origin --delete fix/authentication

<https://www.freecodecamp.org/news/how-to-delete-a-git-branch-both-locally-and-remotely/>

29. How do you configure the command prompt to show the branch?

<https://docs.gitlab.com/ee/gitlab-basics/start-using-git.html>

30. How do you Merge Branches?

<https://www.javatpoint.com/git-merge-and-merge-conflict>

31. Fast-forward merge vs. true merge

<https://medium.com/@koteswar.meesala/git-fast-forward-merge-vs-three-way-merge-8591434dd350>

<https://www.atlassian.com/git/tutorials/using-branches/git-merge>

32. How do you reduce merge conflicts?

Proper branching strategy

Merge asap (at least once a day)

Other common learnings

Prevention rules:

Whenever it is possible, use a new file in preference to an existing one

(the only ambiguity could happen is the same name and path of the file)

Do not always put your changes at the end of a file

(decreases the probability of editing the same line of code)

Do not organise imports

(decreases the probability of editing the same line of code)

Do not beautify a code outside of your changes

(decreases the probability of editing the same line of code)

Push and pull changes as often as you can

(Mitigates distributed nature of Git)

Q. Apache Maven is a software project management and comprehension tool. A build automation tool that helps managing the software build lifecycle.

Q. Maven artifact

An artifact is a JAR, that gets deployed to a Maven repository.

Each artifact has a group ID , an artifact ID (artifact name) and a version string.

Q. Types of maven repository.

local. central. and remote.

Q. Build phases in Maven

Validate. Compile. Test. Package. Install. Deploy.

Q. Explain the highlevel directory structure of a maven project

target: folder holds the compiled unit of code as part of the build process.

<b>Source:</b> folder usually src/main/java holds java source codes.

<b>Test:</b> directory is src/main/test that has all the unit testing codes.

Q. What jar:jar goal does?

It creates the jar files from the target/classes directory without recompiling any source classes.

Q. What are the minimum requirements for a POM?

The minimum requirement for a POM is the following,

Project root modelVersion groupId artifactId version

Q. Can we use different name for POM.xml?

Yes. You could mention file name using the -f option. mvn -f parent-pom.xml

Q. How do you rename a maven project?

1) Rename the project using Eclipse or other IDE.

2) Update the artifactId in your pom.xml

Q. Maven repository central

It is the Maven established repository. For example, your POM specify the dependencies and it is not available in the configured local and the remote repository then maven looks for the resource in Maven Central. Maven provides most of the generic dependency resources at this remote location.

Q. POM

POM (Project Object Model) is the fundamental unit of work. It is an XML file which holds the information about the project and configuration details used to build a project by Maven along with its dependencies.

Q. Why Maven is used?

• Create a jar file • Create war file • Compile code • Unit testing of code • Documenting projects • Reporting

Q. Where do we find .class files of a Maven project?

Under the folder ${basedir}/target/classes/.

Q. Difference between compile and install.

Compile compiles the source code of the project whereas Install installs the package into the local repository, for use as a dependency in other projects locally.

Q. What is a Maven project's fully qualified artifact name?

Order by which Maven searches for the dependency.

Local -> Remote -> Maven Central

groupId - Identifies a project uniquely across all projects.

artifactId - It becomes the name of the jar file.

Q. Explain the difference phases in Maven build Lifecycle.

Validate - validates the project is correct and all necessary information is available.

Compile- compiles the source code of the project.

Test- tests the compiled source code using a suitable unit testing framework. These tests do not require the code to be packaged or deployed.

Package- takes the compiled code and packages it to its distributable format, for example, JAR.

Verify- runs any checks on results of integration tests to ensure desired quality criteria are met.

Install- installs the package into the local repository, for using it as a dependency in other projects locally.

Deploy- performed in the build environment, copies the final package to the remote repository for sharing and collaboration with other members of the team and projects.

Q. How do I specify packaging/distributable format in Maven?

The packaging for your project can be specified via the POM element.

Some of the valid packaging values are jar, war, ear and pom. If no packaging value has been specified, it will default to jar.

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

...

<packaging>war</packaging>

...

</project>

Q. How Maven searches for dependency JAR?

Maven searches first for a dependency JAR in local repository. If found it is used else maven looks at the remote repository or central repository and download the corresponding version of JAR file and then stores it into local repository.

Q. Explain package phase in Maven build lifecycle.

Package phase pulls the compiled code and package it to a distributable format, such as a JAR. The below is the command to package a maven project.

mvn -package

Q. What are the build lifecycles of Maven?

There are 3 built-in build lifecycles.

* The default lifecycle handles your project deployment.
* The clean lifecycle handles project cleaning.
* The site lifecycle handles the creation of the project's site documentation.

Q. Maven Repository

The location where all the project jars, library jars, plugins or any other project related artifacts that are stored and can be easily used by Maven.

Q. What is local repository in Maven?

Maven local repository is located in your local system and is created by the maven when you run any maven command.

By default, maven local repository is %USER\_HOME%/.m2 directory.

We can change the location of maven local repository by changing the settings.xml file. It is located in MAVEN\_HOME/conf.

Q. What is Maven Central Repository?

Maven central repository is located on the web created by the apache maven community.

The path of central repository is: <http://repo1.maven.org/maven2/>.

Q. Explain Maven remote repository

Remote repositories refer to any other type of repository, accessed by a variety of protocols such as file:// and http://. These repositories are set up by a third party to provide their artifacts for downloading.

A "remote" repository may also be an internal repository set up on a file or HTTP server within an organization, used to share private artifacts between development teams and for releases.

Q. How do I include dependencies in a jar using Maven?

Using jar-with-dependencies as the descriptorRef of your assembly-plugin configuration, we can create a JAR along its dependencies. This built-in descriptor creates an assembly with the classifier jar-with-dependencies using the JAR archive format. Below is the subset of pom.xml that includes assembly-plugin configuration along with jar-with-dependencies descriptor. maven-assembly-plugin package single jar-with-dependencies

Configuring resource directories in Maven.

By default, Maven configures and lookup into the src/main/resources directory for your project resources. Also additional resources directories could be specified by adding the configuration to the project's Pom.xml. ... ... src/proj\_resources ... ...

Q. Difference between repository and dependency in Maven.

Repository is a collection of artifacts (eg: jars). You can think of it as a mere storage / cache of various artifacts. Dependency is a situation where your project dependent on another artifact to perform its task. (For example., compile, run, unit test etc.)

Q. What is an Archetype in Maven framework?

Archetype is a Maven plugin that creates a project structure as per its template. The below is the command to create a new maven project based on an archetype.

mvn archetype:generate

Q. What is goal in Maven?

A maven goal represents a specific task that contributes to the building and managing of a Maven project. It may be bound to zero or more build phases. A goal that does not bound to any build phase could be executed outside of the build lifecycle by direct invocation.